CLAIMS

What is claimed is:

1. A method for handling an interrupt in a computer system, the interrupt having a source, the method comprising:

determining if a first-level handler is installed for the interrupt source;

calling said first-level handler if one is installed for the interrupt source;

masking the interrupt source if a first-level handler is not installed for the interrupt source;

calling a second-level handler if a first-level handler is not installed for the interrupt source; and

unmasking the interrupt source if a first-level handler is not installed for the interrupt source.

2. The method of claim 1, further comprising:

retrieving the interrupt source.

3. The method of claim 1, further comprising:

determining if the interrupt source is active; and

returning from the method if the interrupt source is not active.

4. The method of claim 1, further comprising:

setting an interrupt pending flag for the interrupt source before said calling a second-level handler if a first level-handler is not installed for the interrupt source.

5. The method of claim 4, wherein said calling a second-level handler includes:

clearing the pending interrupt flag for the interrupt source.

6. The method of claim 1, wherein said first-level handler is a kernel-level handler.

7. The method of claim 1, wherein said second-level handler is a user-level handler.

8. A method for handling an interrupt in a computer system, the interrupt having a source, the method comprising:

determining if a first-level handler is installed for the interrupt source;

calling said first-level handler if one is installed for the interrupt source;

masking the interrupt source if a first-level handler is not installed for the interrupt source;

determining if a second-level handler is installed for the interrupt source if a first-level handler is not installed for the interrupt source;

calling said second-level handler if a first-level handler is not installed for the interrupt source and a second-level handler is installed for the interrupt source; and

unmasking the interrupt source if a first-level handler is not installed for the interrupt source.

9. The method of claim 8, further comprising:

determining if a third-level handler is installed for the interrupt source if a first-level handler is not installed for the interrupt source and a second-level handler is not installed for the interrupt source; and

calling said third-level handler if a first-level handler is not installed for the interrupt source, a second-level handler is not installed for the interrupt source, and a third-level handler is installed for the interrupt source.

10. The method of claim 8, further comprising:

retrieving the interrupt source.

11. The method of claim 8, further comprising:

determining if the interrupt source is active; and

returning from the method if the interrupt source is not active.

12. The method of claim 8, further comprising:

setting an interrupt pending flag for the interrupt source before said calling said second-level handler.

13. The method of claim 12, wherein said calling a second-level handler includes:

clearing the pending interrupt flag for the interrupt source.

14. An apparatus for handling an interrupt in a computer system, the interrupt having a source, the apparatus comprising:

an installed first-level handler determiner;

a first-level handler caller coupled to said installed first-level handler determiner;

an interrupt source masker coupled to said installed-first level handler determiner;

an installed second-level handler determiner coupled to said interrupt source masker;

a second-level handler caller coupled to said installed second-level handler determiner; and

an interrupt source unmasker coupled to said second-level handler caller.

15. The apparatus of claim 14, further comprising:

an installed third-level handler determiner coupled to said installed second-level handler determiner; and

a third-level handler caller coupled to said installed third-level handler determiner and to said interrupt source unmasker.

16. The apparatus of claim 14, further comprising:

an interrupt source retriever coupled to said installed first-level handler determiner.

17. The apparatus of claim 14, further comprising:

an interrupt source active determiner coupled to said installed first-level handler determiner; and

a method returner coupled to said interrupt source active determiner.

18. The apparatus of claim 14, further comprising:

an interrupt pending flag setter coupled to said interrupt source masker.

19. The apparatus of claim 18, wherein said second-level handler caller includes a pending interrupt flag clearer.

20. An apparatus for handling an interrupt in a computer system, the interrupt having a source, the apparatus comprising:

means for determining if a first-level handler is installed for the interrupt source;

means for calling said first-level handler if one is installed for the interrupt source;

means for masking the interrupt source if a first-level handler is not installed for the interrupt source;

means for calling a second-level handler if a first-level handler is not installed for the interrupt source; and

means for unmasking the interrupt source if a first-level handler is not installed for the interrupt source.

21. The apparatus of claim 20, further comprising:

means for retrieving the interrupt source.

22. The apparatus of claim 20, further comprising:

means for determining if the interrupt source is active; and

means for returning from the method if the interrupt source is not active.

23. The apparatus of claim 20, further comprising:

means for setting an interrupt pending flag for the interrupt source before said calling a second-level handler if a first level-handler is not installed for the interrupt source.

24. The apparatus of claim 23, wherein said means for calling a second-level handler includes:

means for clearing the pending interrupt flag for the interrupt source.

25. The apparatus of claim 20, wherein said first-level handler is a kernel-level handler.

26. The apparatus of claim 20, wherein said second-level handler is a user-level handler.

27. An apparatus for handling an interrupt in a computer system, the interrupt having a source, the apparatus comprising:

means for determining if a first-level handler is installed for the interrupt source;

means for calling said first-level handler if one is installed for the interrupt source;

means for masking the interrupt source if a first-level handler is not installed for the interrupt source;

means for determining if a second-level handler is installed for the interrupt source if a first-level handler is not installed for the interrupt source;

means for calling said second-level handler if a first-level handler is not installed for the interrupt source and a second-level handler is installed for the interrupt source; and

means for unmasking the interrupt source if a first-level handler is not installed for the interrupt source.

28. The apparatus of claim 27, further comprising:

means for determining if a third-level handler is installed for the interrupt source if a first-level handler is not installed for the interrupt source and a second-level handler is not installed for the interrupt source; and

means for calling said third-level handler if a first-level handler is not installed for the interrupt source, a second-level handler is not installed for the interrupt source, and a third-level handler is installed for the interrupt source.

29. The apparatus of claim 27, further comprising:

means for retrieving the interrupt source.

30. The apparatus of claim 27, further comprising:

means for determining if the interrupt source is active; and

means for returning from the method if the interrupt source is not active.

31. The apparatus of claim 27, further comprising:

means for setting an interrupt pending flag for the interrupt source before said calling said second-level handler.

32. The apparatus of claim 31, wherein said means for calling a second-level handler includes:

means for clearing the pending interrupt flag for the interrupt source.

33. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform a method for handling an interrupt in a computer system, the interrupt having a source, the method comprising:

determining if a first-level handler is installed for the interrupt source;

calling said first-level handler if one is installed for the interrupt source;

masking the interrupt source if a first-level handler is not installed for the interrupt source;

calling a second-level handler if a first-level handler is not installed for the interrupt source; and

unmasking the interrupt source if a first-level handler is not installed for the interrupt source.

34. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform a method for handling an interrupt in a computer system, the interrupt having a source, the method comprising:

determining if a first-level handler is installed for the interrupt source;

calling said first-level handler if one is installed for the interrupt source;

masking the interrupt source if a first-level handler is not installed for the interrupt source;

determining if a second-level handler is installed for the interrupt source if a first-level handler is not installed for the interrupt source;

calling said second-level handler if a first-level handler is not installed for the interrupt source and a second-level handler is installed for the interrupt source; and

unmasking the interrupt source if a first-level handler is not installed for the interrupt source.